Introduction to control systems
TI0118

Homework 1

Student name: Pedro Leinos Falcao Cunha

Student number: 542114

Student name: Kelvin Leandro Martins

Student number: 540006

Exercise 1

Find the inverse Laplace transform by hand calculations and verify your results using the Symbolic toolbox for the following functions:

a. $F_1(s) = \dfrac{3s^2 + 5s}{s^3 + 6s^2 + 11s + 6}$

b. $F_2(s) = \dfrac{s^2 + 2s + 1}{(s+2)^3}$

c. $F_3(s) = \dfrac{2s + 3}{s^3 + 6s^2 + 21s + 26}$

d. $F_4(s) = \dfrac{1 + 2e^{-s}}{s^2 + 3s + 2}$

Theoretical basis

According to [LG05], for a signal $x(t)$ in the time-domain, the Laplace transform is defined as

$$X(s) = \mathscr{L}\{x(t)\} = \int_0^\infty x(t)e^{-st}dt$$

where s is a complex variable ($s = \sigma + j\omega$). The Laplace transform is useful to simplify system analysis by transforming time-domain differential equations into easier-to-solve algebraic equations in the s-domain.

The inverse Laplace transform is the process to convert a function (signal) in the s-domain back to the time-domain, defined as

$$x(t) = \mathscr{L}^{-1}\{X(s)\} = \frac{1}{2\pi j}\int_{c-j\infty}^{c+j\infty} X(s)e^{st}ds$$

where c is a constant chosen to guarantee the convergence of the integral. The inverse transformation is essential for converting algebraic solutions back into the time domain, allowing the practical application of theoretical solutions to real-world systems.

For example, let's say that $x(t)$ is the Heaviside step function, popularly denoted by $\theta(t)$, defined as

$$\theta(t) = \begin{cases} 1, & \text{if } t \geq 0 \\ 0, & \text{if } t < 0 \end{cases}$$

Considering $t > 0$, we can calculate the Laplace transformation substituting the function in equation and find $X(s)$ as shown below:

$$X(s) = \mathscr{L}\{x(t)\} = \int_0^\infty x(t)e^{-st}dt = \int_0^\infty e^{-st}dt = \frac{1}{s}$$

So the Heaviside function in the s-domain is $X(s) = \frac{1}{s}$.

Item 1.1

i) Factoring the denominator and using partial fraction decomposition[1]:

$$\begin{aligned} F_1(s) &= \frac{3s^2+5s}{s^3+6s^2+11s+6} \\ &= \frac{3s^2+5s}{(s+1)(s+2)(s+3)} \\ &= \frac{A}{s+1} + \frac{B}{s+2} + \frac{C}{s+3} \end{aligned}$$

ii) Solving the coefficients for A, B, and C:

$$\begin{aligned} A &= \lim_{s\to -1}(s+1)F_1(s) = \lim_{s\to -1} \frac{3s^2+5s}{(s+2)(s+3)} = -1, \\ B &= \lim_{s\to -2}(s+2)F_1(s) = \lim_{s\to -2} \frac{3s^2+5s}{(s+1)(s+3)} = -2, \\ C &= \lim_{s\to -3}(s+3)F_1(s) = \lim_{s\to -3} \frac{3s^2+5s}{(s+1)(s+2)} = 6. \end{aligned}$$

iii) Substituting A, B, and C with their values:

$$F_1(s) = -\frac{1}{s+1} - \frac{2}{s+2} + \frac{6}{s+3}$$

iv) Applying the inverse Laplace transform:

$$\begin{aligned} \mathscr{L}^{-1}\{F_1(s)\} &= \mathscr{L}^{-1}\left\{-\frac{1}{s+1} - \frac{2}{s+2} + \frac{6}{s+3}\right\} \\ &= -e^{-t}u(t) - 2e^{-2t}u(t) + 6e^{-3t}u(t). \end{aligned}$$

Using *SymPy* for symbolic computations (Listing 1), we get the following result for the inverse Laplace transform:

$$f_1(t) = \left(-e^{2t} - 2e^{t} + 6\right) e^{-3t}\theta\left(t\right)$$

where $\theta(t)$ denotes the unit step function.

[1] we use the partial fraction decomposition to separate each component of F(s) and make it easier to convert them from the s-domain to the time domain.

Item 1.2

i) Using the method of partial fraction decomposition:

$$\begin{aligned} F_2(s) &= \frac{s^2+2s+1}{(s+2)^3} \\ &= \frac{A}{(s+2)^3} + \frac{B}{(s+2)^2} + \frac{C}{(s+2)} \end{aligned}$$

ii) Solving the coefficients for A, B, and C:

$$\begin{aligned} A &= \lim_{s\to -2} \frac{1}{0!} \frac{d^0[(s+2)^3 F_2(s)]}{ds^0} = \lim_{s\to -2} (s^2+2s+1) = 1, \\ B &= \lim_{s\to -2} \frac{1}{1!} \frac{d[(s+2)^3 F_2(s)]}{ds} = \lim_{s\to -2} (2s+2) = -2, \\ C &= \lim_{s\to -2} \frac{1}{2!} \frac{d^2[(s+2)^3 F_2(s)]}{ds^2} = \lim_{s\to -2} 1 = 1. \end{aligned}$$

iii) Substituting A, B, and C with their calculated values:

$$F_2(s) = \frac{1}{(s+2)^3} - \frac{2}{(s+2)^2} + \frac{1}{s+2}$$

iv) Applying the inverse Laplace transform:

$$\begin{aligned} \mathscr{L}^{-1}\{F_2(s)\} &= \mathscr{L}^{-1}\left\{\frac{1}{(s+2)^3} - \frac{2}{(s+2)^2} + \frac{1}{s+2}\right\} \\ &= \frac{t^2 e^{-2t}}{2} u(t) - 2te^{-2t}u(t) + e^{-2t}u(t) \end{aligned}$$

Calculating the inverse Laplace transform with SymPy (Listing 2), we get the following result for $f(t)$:

$$f_2(t) = \frac{\left(t^2-4t+2\right) e^{-2t}\theta\left(t\right)}{2}$$

which is the same result found manually, but it‘s simplified.

Item 1.3

i) Factoring the denominator and partial fraction decomposition:

$$\begin{aligned}
F_3(s) &= \frac{2s+3}{s^3+6s^2+21s+26} \\
&= \frac{2s+3}{(s+2)(s^2+4s+13)} \\
&= \frac{A}{s+2} + \frac{Bs+C}{s^2+4s+13} \\
&= \frac{A(s^2+4s+13)+(Bs+C)(s+2)}{s^3+6s^2+21s+26} \\
&= \frac{(A+B)s^2+(4A+2B+C)s+(13A+2C)}{s^3+6s^2+21s+26}
\end{aligned}$$

ii) Solving the linear system for A, B, and C [3]:

$$\begin{bmatrix} 1 & 1 & 0 \\ 4 & 2 & 1 \\ 13 & 0 & 2 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 3 \end{bmatrix} \rightarrow \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} \frac{-1}{9} \\ \frac{1}{9} \\ \frac{20}{9} \end{bmatrix}$$

iii) Substituting A, B, and C with their calculated values:

$$\begin{aligned}
F_3(s) &= -\frac{1}{9}\frac{1}{(s+2})) + \frac{1}{9}\frac{s+20}{(s^2+4s+13)} \\
&= -\frac{1}{9}\frac{1}{(s+2)} + \frac{1}{9}\left[\frac{s+2}{(s+2)^2+3^2} + 6\frac{3}{(s+2)^2+3^2}\right]
\end{aligned}$$

iv) Applying the inverse Laplace transform:

$$\begin{aligned}
\mathscr{L}^{-1}\{F_3(s)\} &= -\frac{1}{9}\mathscr{L}^{-1}\left\{\frac{1}{s+2}\right\} + \frac{1}{9}\mathscr{L}^{-1}\left\{\frac{s+2}{(s+2)^2+3^2} + 6\frac{3}{(s+2)^2+3^2}\right\} \\
&= -\frac{e^{-2t}}{9}u(t) + \frac{1}{9}\left[e^{-2t}\cos(3t) + 6e^{-2t}\sin(3t)\right]u(t)
\end{aligned}$$

Using SymPy (Listing 3), the inverse Laplace transform is:

$$f_3(t) = \frac{(6\sin(3t) + \cos(3t) - 1)\, e^{-2t}\theta(t)}{9}$$

Putting the term e^{-2t} in evidence in the result found manually, it becomes the same result found with SymPy.

[3] Since the denominator of F(s) has complex roots, we factor the polynom and calculate the respective coefficients, and them compare them to the initial F(s).

Item 1.4

i) Factoring the denominator and setting up for partial fractions:

$$
\begin{aligned}
F_4(s) &= \frac{1+2e^{-s}}{s^2+3s+2} \\
&= \frac{1+2e^{-s}}{(s+1)(s+2)} \\
&= \frac{1}{(s+1)(s+2)} + 2e^{-s}\frac{1}{(s+1)(s+2)} \\
&= \frac{1}{s+1} + \frac{2}{(s+1)}e^{-s} - \frac{1}{s+2} - \frac{2}{(s+2)}e^{-s}
\end{aligned}
$$

ii) Applying the inverse Laplace transform:

$$
\begin{aligned}
\mathscr{L}^{-1}\{F_4(s)\} &= \mathscr{L}^{-1}\left\{\frac{1}{s+1} + \frac{2}{(s+1)}e^{-s} - \frac{1}{s+2} - \frac{2}{(s+2)}e^{-s}\right\} \\
&= e^{-t}u(t) + 2e^{-(t-1)}u(t-1) - e^{-2t}u(t) - 2e^{-2(t-1)}u(t-1)
\end{aligned}
$$

Using SymPy (see Listing 4), the inverse Laplace transform in its expanded form is:

$$
f_4(t) = e^{-t}\theta\left(t\right) + 2ee^{-t}\theta\left(t-1\right) - e^{-2t}\theta\left(t\right) - 2e^{2}e^{-2t}\theta\left(t-1\right)
$$

which is the same result found manually.

Exercise 2

In order to study the effect of zeros in response of a system, consider the transfer function given in equation (1).

$$G(s) = \frac{\alpha s + 1}{2s^2 + 3s + 1} \tag{1}$$

Do the following:

1. For $\alpha = 1$, find and plot the unit step and impulse responses.
2. For $\alpha = [-1,\ 5,\ 0,\ 5\ ,7]$, plot and compare the unit step response.
3. Discuss how the system varies its response for the different values of α.

Theoretical basis

Transfer function is a mathematical model that relates the system response $Y(s)$ to an input signal $X(s)$ when the initial conditions are null. We can find the transfer function $H(s)$ dividing $Y(s)$ by $X(s)$:

$$H(s) = \frac{Y(s)}{X(s)} = \frac{\mathscr{L}\{y(t)\}}{\mathscr{L}\{x(t)\}} = \frac{Num(s)}{Den(s)} = \frac{b_m s^m + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0}$$

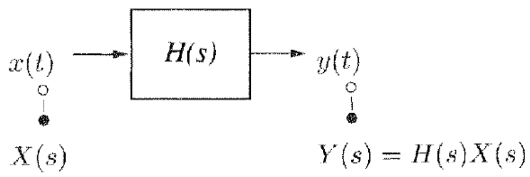

Figure 1: Determining the system response with the transfer function.

Poles and *zeros* are the roots of the denominator and numerator, respectively, of a transfer function. Since in the s-domain s is a complex variable, the roots can also assume complex values, as explained by [GRS01].

We can determine the order of the system by looking at the number of poles, but the number of zeros does not influence the system's order. The poles primarily affect the system's stability and the transient response. A system is stable if its poles are in the left half of the complex plane (continuous systems) or inside the unit circle (discrete systems), ensuring the output will not diverge over time. The transient response describes how the system reacts from an initial state to a change before reaching a steady state, influenced by the system's poles. The closer a pole is to the imaginary axis, the slower the system's response will decay over time. Zeros influence the shape of the system's response but not its stability.

Item 2.1

i) Factoring the denominator:

$$\begin{aligned} G(s) &= \frac{\alpha s + 1}{2s^2 + 3s + 1} \\ &= \frac{\alpha s + 1}{(s+1)(2s+1)} \end{aligned}$$

ii.i) For $u(t) = \delta(t)$ and $\alpha = 1$:

$$\begin{aligned} Y(s) &= G(s)U(s) = G(s) \\ &= \frac{s+1}{(s+1)(2s+1)} = \frac{1}{2s+1} \end{aligned}$$

ii.ii) Applying the inverse Laplace transform:

$$Y(s) = \frac{1}{2s+1} \xrightarrow{\mathscr{L}^{-1}\{Y(s)\}} y(t) = \frac{e^{-t/2}}{2}$$

iii.i) For $u(t) = \theta(t)$:

$$\begin{aligned} Y(s) &= G(s)U(s) = G(s) \\ &= \frac{\alpha s + 1}{(s+1)(2s+1)s} = \frac{A}{s+1} + \frac{B}{2s+1} + \frac{C}{s} \end{aligned}$$

iii.ii) Solving the coefficients for A, B, and C:

$$\begin{aligned} A &= \lim_{s \to -1} (s+1)Y(s) = \lim_{s \to -1} \frac{\alpha s + 1}{(2s+1)s} = -\alpha + 1, \\ B &= \lim_{s \to -1/2} (2s+1)Y(s) = \lim_{s \to -1/2} \frac{\alpha s + 1}{(s+1)s} = 2\alpha - 4, \\ C &= \lim_{s \to 0} sY(s) = \lim_{s \to 0} \frac{\alpha s + 1}{(s+1)(2s+1)} = 1. \end{aligned}$$

iii.iii) Substituting A, B, and C with their values:

$$Y(s) = \frac{-\alpha + 1}{s+1} + \frac{2\alpha - 4}{2s+1} + \frac{1}{s}$$

iii.iv) Applying the inverse Laplace transform:

$$Y(s) \xrightarrow{\mathscr{L}^{-1}\{Y(s)\}} y(t) = (-\alpha + 1)e^{-t} + (2\alpha - 4)e^{-t/2} + 1$$

iii.v) For $\alpha = 1$:

$$y(t) = -e^{-t/2} + 1$$

Creating the transfer function and generating the impulse response with *SciPy* (Listing 5), the response generated is shown below:

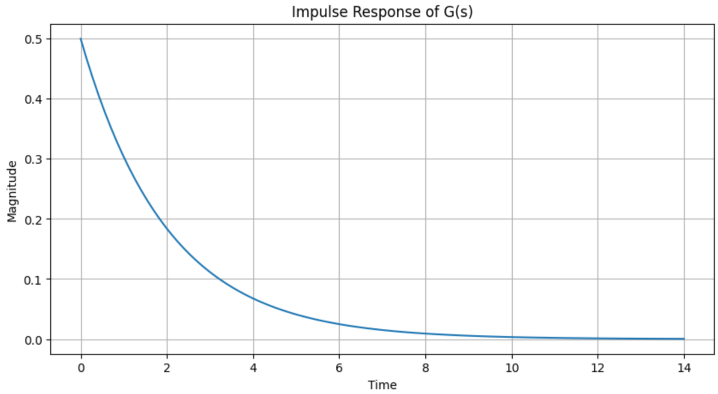

Figure 2: Impulse response of $G(s)$

Similarly, using the unit step function as input, the response below was generated:

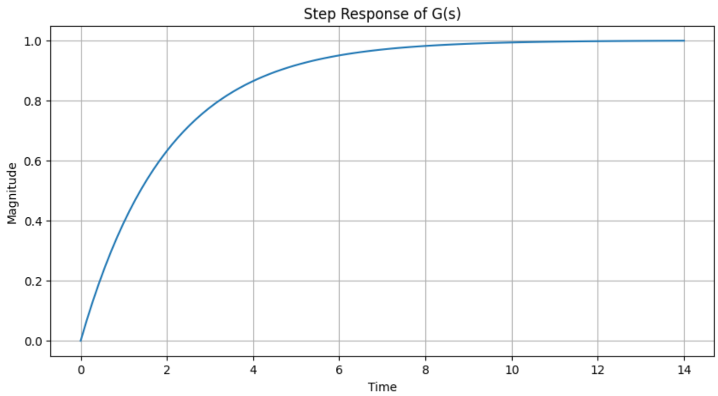

Figure 3: Impulse response of $G(s)$

The impulse response in Figure 2 shows that the system's response decays to zero relatively quickly, and the curve stabilizes without any oscillations. This indicates that the system effectively dissipates any effects of the initial impulse over time.

The step response graph in Figure 3 shows that after an initial rise, the system's response stabilizes and levels off, approaching a steady state. This behavior demonstrates that the system can handle a step input without diverging, further supporting the notion of stability. The poles of the system, calculated from the denominator of the transfer function, are $x_1 = -1$ and $x_2 = -0.5$. Since both poles have negative real parts and no imaginary parts, they lie in the left half of the complex plane. This confirms that the system is stable.

Item 2.2

With the code in Listing 6, the step response is plotted to observe how it differs when each α is used, demonstrating the system's sensitivity to this parameter. Each plot reveals distinct dynamics, such as stabilization and overshoot behaviors. The comparative analysis can be visualized in Figure 4, which showcases the changes in system response across the different α values.

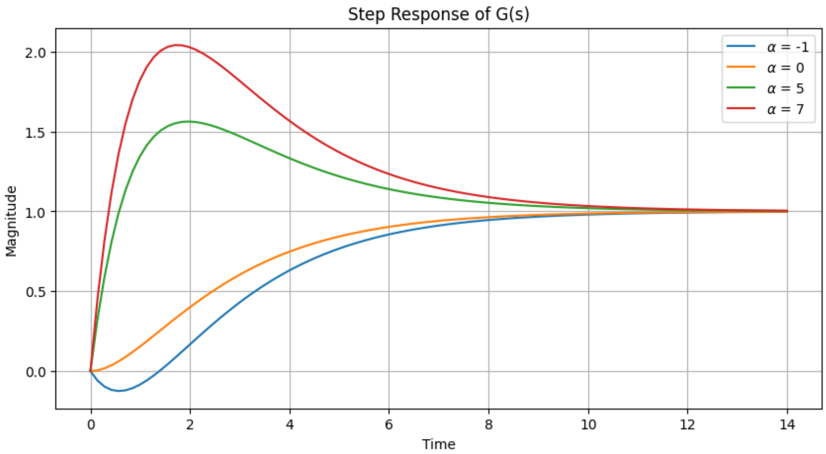

Figure 4: Step response varying α

Item 2.3

The system's sensitivity and dynamic response are directly influenced by the magnitude and sign of α. A negative α causes an initial negative response, stabilizing above 0.5. Zero α results in a smooth rise towards 0.5. Positive α values produce a more vigorous and rapid response, reaching higher peaks before settling.

The zeros in this case are straightforward to determine. Given that the numerator is a linear polynomial, each zero is given by $-1/\alpha$. However, when $\alpha = 0$, there is no zero. To better illustrate the poles and zeros positions, the pole-zero plot was generated in the figure below.

The zero's location relative to the imaginary axis in the s-plane plays a pivotal role in shaping the transient dynamics. Zeros on the left (positive α) tend to enhance the system's initial responsiveness and overshoot, whereas zeros on the right (negative α) can cause an initial opposing effect, leading to undershoot before recovery. The case where $\alpha = 0$, the numerator will be a constant. This will consequently remove the zero's influence, and the system's response starts from zero and rises smoothly towards the steady state without any overshoot or undershoot.

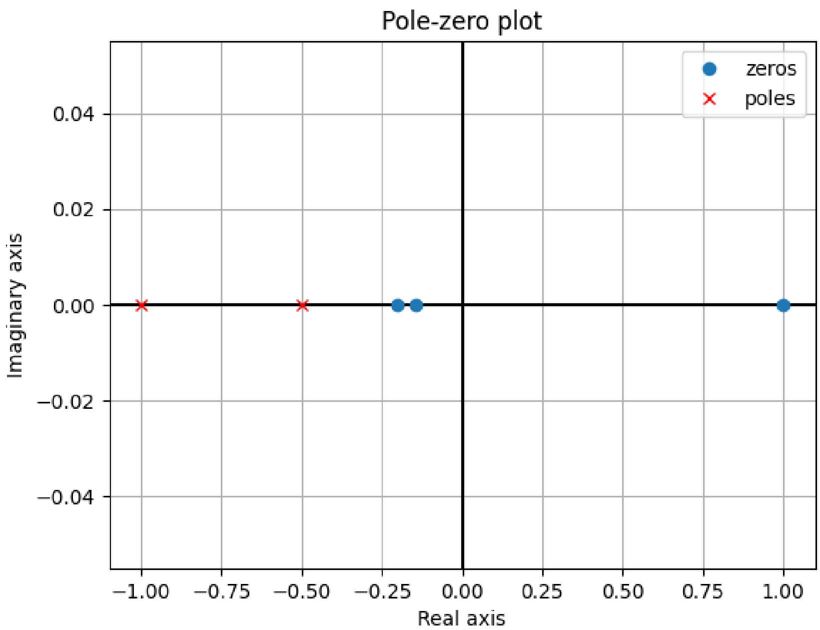

Figure 5: Pole-zero plot

Exercise 3

Consider the input-output model in equation (2):

$$\frac{d^2y(t)}{dt^2} + 2\frac{dy(t)}{dt} + y(t) = 4\frac{d^2u(t)}{dt} + 15\frac{du(t)}{dt} + 19u(t) \tag{2}$$

1. Define the characteristic polynomial and plot the system modes.
2. Given the initial conditions in equation (3), find the free evolution of the system in equation (2):
$$y(t)\Big|_{t=0} = 1, \quad \frac{dy(t)}{dt}\Big|_{t=0} = 1 \tag{3}$$
3. Find the forced response of the system subject to a unit step input.
4. By means of the programming language of your choice, plot the response $y(t)$ and comment on your results.

Theoretical basis

An nth-order linear differential equation has the form

$$b_n(x)y^n + b_{n-1}(x)y^{n-1} + \cdots + b_0(x)y = g(x)$$

where the superscript *(n)* denotes the n^{th}-derivative, and each b_j are constants. $b_j(x)$ and $g(x)$ does not depend on y or its derivatives. So, they depend only on the variable x. If $g(x)$ is equal to zero, we say it's a *homogeneous* equation, otherwise, is a *nonhomogeneous* equation. A general solution of a equation is the sum of the homogeneous and the particular (nohomogeneous) solutions.

The total response of the system is given by sum of the zero-input response and the zero-state response. The *zero-input response* (or free response) is system's response when $x(t) = 0$, so this is result of the internal conditions of the system alone, or in other words, it does not depend on the input. The *zero-state response* (or forced response) is the output of a system when the initial conditions are set to zero and the system is stimulated by an input signal. This response shows how the system reacts to external inputs.

According to [Bro03], to find the characteristic equation, we assume that the solution has the form $y = e^{\lambda t}$, where λ is a constant to be determined. So we replace y for $e^{\lambda t}$ in the differential equation. With this, each derivative of y becomes $\lambda^k e^{\lambda t}$, where k is the order of the derivative. After the substitution, we obtain a polynomial equation in λ, and this is called the *characteristic equation* or *characteristic polynomial*. This polynomial is crucial for determining the system's behavior. It allows us to find specific solutions that satisfy the initial conditions of the system.

As [Baz21] exemplifies, let's assume a generic homogeneous ODE, as the one shown below:

$$a\frac{d^2y}{dt^2} + b\frac{dy}{dt} + cy = 0$$

Replacing y for $e^{\lambda t}$ we obtain:

$$a\frac{d^2(e^{\lambda t})}{dt^2} + b\frac{d(e^{\lambda t})}{dt} + ce^{\lambda t} = 0$$

Calculating the derivatives and then simplify it, we get the following result:

$$a\lambda^2 e^{\lambda t} + b\lambda e^{\lambda t} + ce^{\lambda t} = 0$$

$$e^{\lambda t}(a\lambda^2 + b\lambda + c) = 0 \to a\lambda^2 + b\lambda + c = 0$$

For this quadratic equation, let's say the roots are λ_1 and λ_2. The roots of the characteristic equation determine the solution of the differential equation. The general solution for the second-order equation is:

$$y(t) = c_1 e^{\lambda_1 t} + c_2 e^{\lambda_2 t}$$

Assuming a generic N^{th} order differential equation, the characteristic equation $Q(\lambda)$ can be expressed as

$$Q(\lambda) = (\lambda - \lambda_1)(\lambda - \lambda_2) \cdots (\lambda - \lambda_N) = 0$$

Consequently, the equation will also have N possible solutions, expressing as:

$$y(t) = c_1 e^{\lambda_1 t} + c_1 e^{\lambda_1 t} + \cdots + c_N e^{\lambda_N t}$$

[LG05] states that each exponential in the previous equation from the zero-input response is called the *characteristic modes* of the system. For each root found in the characteristic equation there will be a mode. The zero-input response (also called free response) is a linear combination of the characteristic modes. These modes not only determine the free response but also it's important to find determine the forced response. So the entire behavior of the system is dictated by its characteristic modes.

There are some details to take care about when the roots are complex or repeated. We'll focus only on repeated roots case, because it will be useful in Item 3.1. In case the root is repeated twice in a quadratic equation, for example, the solution will take the form:

$$y(t) = (c_1 + c_2 t)e^{\lambda t}$$

In this case, $e^{\lambda t}$ and $te^{\lambda t}$ are the characteristic modes of the system.

Item 3.1

i) Assuming that the solution has the form $y = e^{\lambda t}$ in the ODE:

$$\lambda^2 e^{\lambda t} + 2\lambda e^{\lambda t} + e^{\lambda t} = (\lambda + 1)^2 \lambda t = 0$$

ii) Therefore, the roots are:

$$\lambda_1 = \lambda_2 = -1$$

iii) So, the solution for the 2nd order equation is:

$$y(t) = c_1 e^{-t} + c_2 t e^{-t}$$

iv) Finally , the characteristics modes of the systems are e^{-t} and te^{-t}, plotted below:

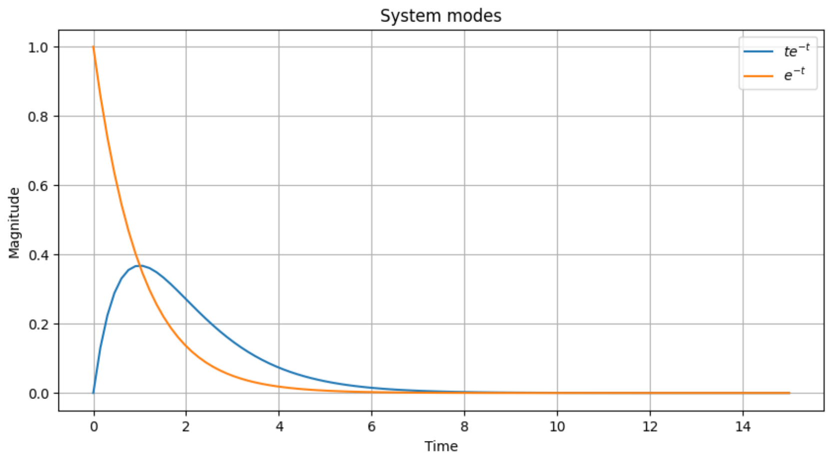

Figure 6: System modes

The first mode (e^{-t}) decays exponentially over time, reflecting a natural decay without any additional factors affecting its rate. The second mode (te^{-t}) initially increases due to the t factor, which effectively counters the exponential decay for a short period, but eventually, the decay e^{-t} dominates, causing the overall value to decrease.

Item 3.2

i) Applying the Laplace transform ($\mathscr{L}$) to both sides of the equation:

$$(s^2Y(s) - sy(0) - \dot{y}(0)) + 2(sY(s) - y(0)) + Y(s) = 4s^2U(s) + 15sU(s) + 19U(s)$$

ii) After rearranging the equation:

$$Y(s) = \frac{sy(0) + 2y(0) + \dot{y}(0)}{s^2 + 2s + 1} + \frac{4s^2 + 15s + 19}{s^2 + 2s + 1}U(s)$$

$$\begin{cases} \text{Free evolution of the system: } \dfrac{sy(0) + 2y(0) + \dot{y}(0)}{s^2 + 2s + 1} \end{cases}$$

$$\begin{cases} \text{Forced evolution of the system: } \dfrac{4s^2 + 15s + 19}{s^2 + 2s + 1}U(s) \end{cases}$$

iii) Finding the free evolution of the system using the initial conditions of equation (3):

$$Y(s)_{free} = \frac{s + 3}{s^2 + 2s + 1}$$

iv) Applying the Inverse Laplace Transform, we find the following free response in time domain:

$$y(t)_{free} = 2te^{-t} + e^{-t} \tag{4}$$

And by plotting this function we get the result in Figure 7:

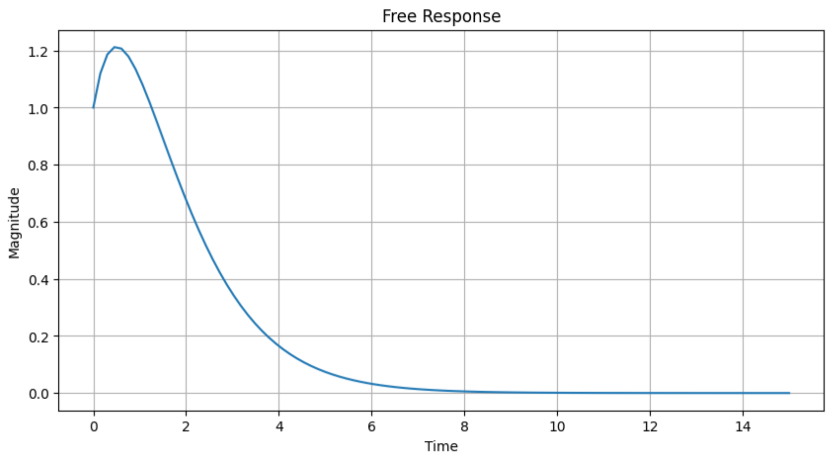

Figure 7: Free response

Item 3.3

i) For $u(t) = \theta(t)$:

$$F(s) = \frac{4s^2 + 15s + 19}{(s+1)^2 s} = \frac{A}{s} + \frac{B}{(s+1)^2} + \frac{C}{(s+1)}$$

ii) Solving the coefficients for A, B, and C:

$$A = \lim_{s \to 0} \frac{4s^2 + 15s + 19}{(s+1)^2} = \lim_{s \to 0} \frac{19}{1} = 19,$$
$$B = \lim_{s \to -1} (s+1)^2 F(s) = \lim_{s \to -1} \frac{4s^2 + 15s + 19}{s} = -8$$
$$C = \lim_{s \to -1} \frac{1}{1!} \frac{d[(s+1)^2 F(s)]}{ds} = \lim_{s \to -1} \frac{4s^2 - 19}{s^2} = -15.$$

ii) Substituting A, B, and C with their values:

$$F(s) = \frac{19}{s} - \frac{8}{(s+1)^2} - \frac{15}{s+1}$$

iii) Applying the Inverse Laplace Transform:

$$f(t) = 19 - 8te^{-t} - 15e^{-t} \tag{5}$$

Plotting this function we get the result below:

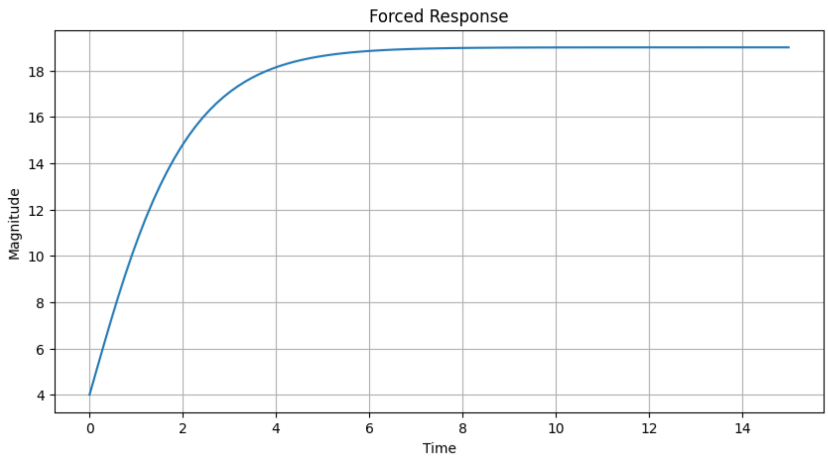

Figure 8: Forced response

ITEM 3.4

Considering the initial conditions given in item 3.2 and the unit step input from item 3.3, the resulted $Y(s)$ is

$$Y(s) = \frac{5s + 18 + \frac{19}{s}}{s^2 + 2s + 1}$$

And applying the Inverse Laplace Transform we find that y(t) is

$$y(t) = 19 - 6te^{-t} - 14e^{-t} \tag{6}$$

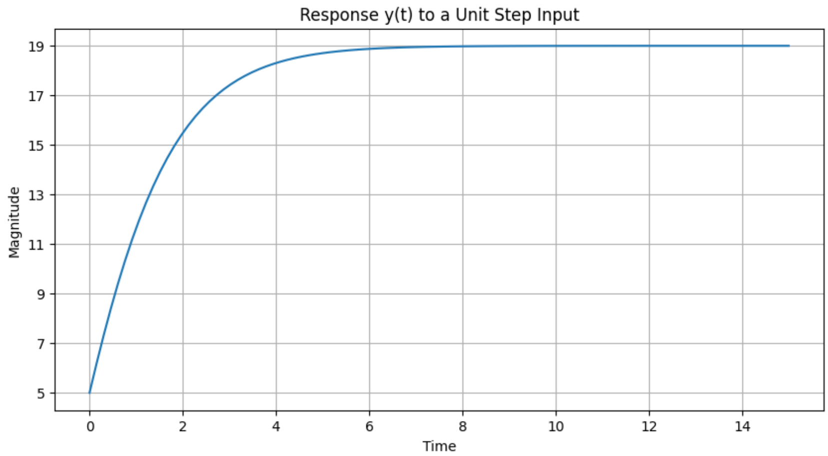

Figure 9: Response of G(s)

The total system response, $y(t)$, as depicted in Figure 9 and defined by Equation 6, results from the algebraic addition of the free and forced responses (Equations 4 and 5, respectively). The plot visually confirms this by illustrating how the sum of the dynamics presented in Figures 7 and 8 shapes the overall trajectory of $y(t)$. This demonstrates the principle of superposition, where the total response is directly influenced by the individual characteristics of both the free and forced responses.

Exercise 4

Given the state-space model in (7):

$$
\begin{cases}
\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} & = \begin{bmatrix} 0 & 1 \\ -5 & -2 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 2 \end{bmatrix} u(t) \\
\\
y(t) & = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + u(t)
\end{cases}
\tag{7}
$$

1. Find a the corresponding transfer function $G(s) = Y(s)/U(s)$.
2. Find an input-output model equivalent to the state-space model.
3. Find the state and output forced evolution as response of the input $u(t) = e^{-3t}\theta(t)$.

Theoretical basis

When we are describing a system there are many ways to achieve this. One of them is the *input-output description*. In this model, we describe the system in terms of its inputs and outputs. The output depends on the input and some internal factors (the system itself). [LG05] contends that the system is governed by laws of interconnection. So, when we are building the system model, or in other words, creating a mathematical expression that describes the dynamical behavior of the system, we use these laws to relate the input to the output.

Even though we examine the internal structure of a system, such as using meshes for an electrical system, this I/O model represents an external description of the system. An external description encompasses all kinds of descriptions that we can obtain by measuring the external terminals (e.g., the input and output current/voltage). However, with an internal description, it's possible to provide details about every signal within the system. With these internal descriptions, we are capable of deriving an external description. For example, consider a mass-spring system as a mechanical system. The internal description of this system involves specifying the mass of the object, the stiffness of the spring, and the damping coefficient (if any). By knowing these internal parameters, we can describe the dynamics of the system, such as its natural frequency of oscillation and how it responds to external forces.

A *state-space* description is an internal description of the system. The states variables are keys variables in a system, and knowing theses variables we can determine every possible output of the system that we may want to measure an/or control.

Of all possible system variables in a specific system, we select a linearly independent subset and call this *state variables*. If this system is an nth-order system, we will write n first order differential equations in terms of these variables, and this will be the *state equations*. Combining mathematically the states variables with the system's input, we can find others systems variables. We call this the *output equation*.

Let's use the *mass-spring damper system* as example as shown below:

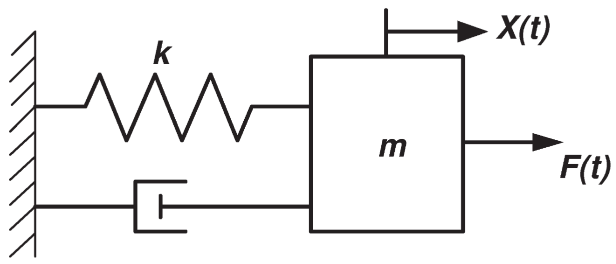

Figure 10: The Mass-Spring Damper System

The system can be described by two state variables.

1. The position x: The displacement of the mass from its equilibrium position.
2. The velocity v: The rate of change of the mass's position.

These state variables capture the entire internal state of the system at any given time. The state-space equations for this system can be written as a set of differential equations:

$$\begin{cases} \dot{x}(t) & = v \\ \dot{v}(t) & = \frac{-kx}{m} + \frac{-bv}{m} \end{cases}$$

In this example, $\dot{x}(t)$ represents the rate of change of position (velocity), and $\dot{v}(t)$ represents the rate of change of velocity (acceleration). The terms $\frac{-kx}{m}$ and $\frac{-bv}{m}$ represent the forces acting on the mass due to the spring and damper, respectively. Knowing these state variables at a specific time (initial conditions), we can use the state-space equations to predict the system's future behavior. One possible output might be the position of the mass in a future time, and we can achieve this solving the differential equations with the initial values of x and v.

It is possible to write the state and output equations in vector-matrix form if the system is linear. As [Nis11] states, the state equations can be written as

$$\dot{x} = \mathbf{A}x + \mathbf{B}u$$

and the output equation can be written as show below:

$$y = \mathbf{C}x + \mathbf{D}u$$

where $\mathbf{u}$ represents the input, $\mathbf{x}$ the state variables vector, $\dot{\mathbf{x}}$ the derivative of the state vector with respect to time, and $\mathbf{y}$ the output.

Having a state-space model we can convert it to a transfer function, or the inverse. Given the state and output equations above, we take the Laplace transform and we assume zero initial conditions:

$$s\mathbf{X}(s) = \mathbf{A}\mathbf{X}(s) + \mathbf{B}\mathbf{U}(s)$$
$$\mathbf{Y}(s) = \mathbf{C}\mathbf{X}(s) + \mathbf{D}\mathbf{U}(s)$$

Isolating *X(s)* from the first equation we get:

$$\mathbf{X}(s) = (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}\mathbf{U}(s) \tag{8}$$

where $\mathbf{I}$ is the identity matrix. And replacing this in output equation, we have the following result:

$$\mathbf{Y}(s) = \left[\mathbf{C}\left(s\mathbf{I} - \mathbf{A}\right)^{-1}\mathbf{B} + \mathbf{D}\right]\mathbf{U}(s)$$

Since $G(s) = Y(s)/U(s)$, the transfer function is:

$$G(s) = \frac{Y(s)}{U(s)} = \left[\mathbf{C}\left(s\mathbf{I} - \mathbf{A}\right)^{-1}\mathbf{B} + \mathbf{D}\right] \tag{9}$$

Item 4.1

i) Using Equation (9):

$$\begin{aligned}
G(s) &= \left[C\,(sI - A)^{-1}\,B + D\right] \\
&= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} s & -1 \\ 5 & s+2 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 2 \end{bmatrix} + \begin{bmatrix} 1 \end{bmatrix} \\
&= \frac{\left(\begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} s+2 & 1 \\ -5 & s \end{bmatrix} \begin{bmatrix} 0 \\ 2 \end{bmatrix}\right)}{s^2 + 2s + 5} + 1 \\
&= \frac{s^2 + 4s + 5}{s^2 + 2s + 5}
\end{aligned}$$

ii) Therefore, the transfer function is:

$$G(s) = \frac{s^2 + 4s + 5}{s^2 + 2s + 5} \tag{10}$$

Using *SciPy*'s function *ss2tf*, it is possible to convert a state-space model to a transfer function. Using the same matrices provided in the exercise, the computational result using the code in Listing 12 was the same as the manual. The output result was this object representing the transfer function:

```
Output:
TransferFunctionContinuous(
array([1., 4., 5.]),
array([1., 2., 5.]),
dt: None
)
```

where the first array is the numerator coefficients from the highest order to the lowest, so this array says the numerator is $s^2 + 4s + 5$, and the second array it's the denominator array, or $s^2 + 2s + 5$.

ITEM 4.2

i) Since $G(s) = Y(s)/U(s)$, from equation (10):

$$
\begin{aligned}
(s^2 + 2s + 5)Y(s) &= (s^2 + 4s + 5)U(s) \\
&\rightarrow s^2Y(s) + 2sY(s) + 5Y(s) = s^2U(s) + 4sU(s) + 5U(s)
\end{aligned}
$$

ii) Finally, using the Inverse Laplace Transform to get the input-output model:

$$
\frac{d^2y(t)}{dt^2} + 2\frac{dy(t)}{dt} + 5y(t) = \frac{d^2u(t)}{dt^2} + 4\frac{du(t)}{dt} + 5u(t)
$$

Item 4.3

i) For $u(t) = e^{-3t}\theta(t)$, $U(s) = \frac{1}{s+3}$. Also, since $Y(s) = G(s)U(s)$, $G(s)$ from equation (10):

$$\begin{aligned} Y(s) &= \frac{s^2+4s+5}{(s^2+2s+5)(s+3)} \\ &= \frac{A}{s+3} + \frac{Bs+C}{s^2+2s+5} \\ &= \frac{(A+B)s^2 + (2A+3B+C)s + (5A+3C)}{(s^2+2s+5)(s+3)} \end{aligned}$$

ii) Solving the linear system for A, B, and C:

$$\begin{bmatrix} 1 & 1 & 0 \\ 2 & 3 & 1 \\ 5 & 0 & 3 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \\ 5 \end{bmatrix} \rightarrow \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} \frac{1}{4} \\ \frac{3}{4} \\ \frac{5}{4} \end{bmatrix}$$

iii) Substituting A, B, and C with their calculated values:

$$\begin{aligned} Y(s) &= \frac{1}{4}\frac{1}{(s+3)} + \frac{1}{4}\frac{3s+5}{(s^2+2s+5)} \\ &= \frac{1}{4}\frac{1}{(s+3)} + \frac{1}{4}\left[3\frac{s+1}{(s+1)^2+2^2} + \frac{2}{(s+1)^2+2^2}\right] \end{aligned}$$

iv) Finally, applying the Inverse Laplace Transform:

$$y(t) = \frac{e^{-3t}}{4} + \frac{1}{4}\left[3e^{-t}\cos(2t) + e^{-t}\sin(2t)\right]$$

For the states response, we'll use the Equation 8.
i) Calculating $(sI-A)^{-1}B$:

$$\begin{aligned} (sI-A)^{-1}B &= \begin{bmatrix} s & -1 \\ 5 & s+2 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 2 \end{bmatrix} \\ &= \frac{1}{s^2+2s+5} \begin{bmatrix} s+2 & -5 \\ 1 & s \end{bmatrix}^T \begin{bmatrix} 0 \\ 2 \end{bmatrix} \\ &= \frac{1}{s^2+2s+5} \begin{bmatrix} 2 \\ 2s \end{bmatrix} \\ &= \begin{bmatrix} \frac{2}{s^2+2s+5} \\ \frac{2s}{s^2+2s+5} \end{bmatrix} \end{aligned}$$

ii) Then, we multiply this matrix we found by $U(s)$:

$$X(s) = \begin{bmatrix} \frac{2}{s^2+2s+5} \\ \frac{2s}{s^2+2s+5} \end{bmatrix} \frac{1}{s+3}$$

iii) With this we find x_1 and x_2 in Laplace domain:

$$X(s) = \begin{bmatrix} X_1(s) \\ X_2(s) \end{bmatrix} = \begin{bmatrix} \frac{2}{s^2+2s+5} \frac{1}{s+3} \\ \frac{2s}{s^2+2s+5} \frac{1}{s+3} \end{bmatrix}$$

iv) Calculating the inverse Laplace transform for each variable:

$$\begin{aligned} x_1(t) = \mathscr{L}^{-1}\{X_1(s)\} &= \mathscr{L}^{-1}\left\{\frac{2}{s^2+2s+5} \cdot \frac{1}{s+3}\right\} \\ &= \frac{(-\sqrt{2}e^{2t}cos(2t+\pi/4)+1)e^{-3t}\theta(t)}{4} \\ x_2(t) = \mathscr{L}^{-1}\{X_2(s)\} &= \mathscr{L}^{-1}\left\{\frac{2s}{s^2+2s+5} \cdot \frac{1}{s+3}\right\} \\ &= \frac{((sin(2t)+3cos(2t))e^{2t}-3)e^{-3t}\theta(t)}{4} \end{aligned}$$

Using Python *control* library, we defined the same state-space model and generated the forced response with the input $u(t) = e^{-3t}\theta(t)$ (Listing 14). The output and states variables evolution are shown in Figure 11.

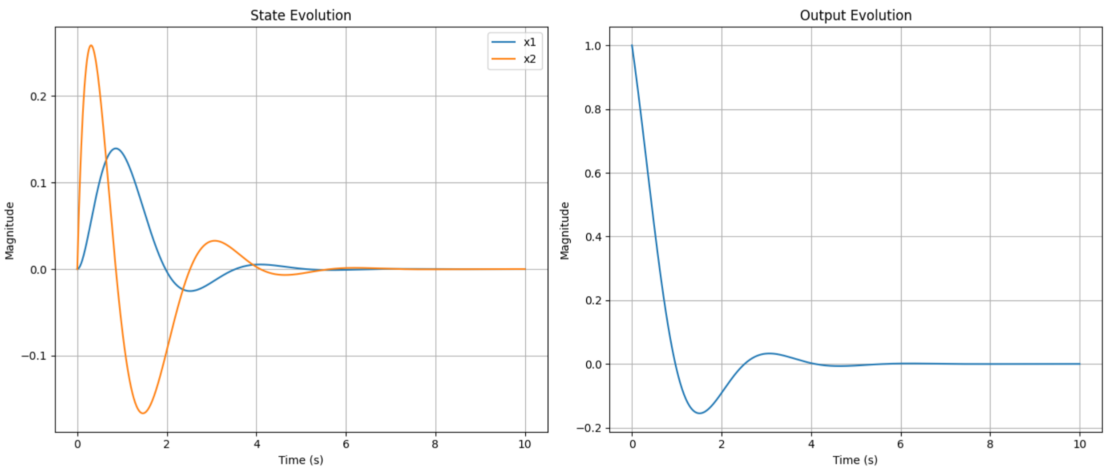

Figure 11: Output and state evolution

The state x_1 shows a peak and then some oscillations before settling down, while the state x_2 shows a larger initial response before also settling. The oscillatory nature of the response suggests that the system is underdamped, which means it has complex-conjugate poles.
Looking at the matrix **C**, that represents the contribution of the system states to the output, we see that the output is directly proportional to the second state variable x_2. This implies that the first state x_1 does not directly influence the output. The matrix

D, represents the direct feedthrough of the input *u(t)* to the output, indicates a direct and unmodified contribution of the input to the output.
The initial peak in the output is higher than the peaks in the state variables, which is due to the direct feedthrough term $\mathbf{D}u(t)$. This results in the output having a significant immediate response to the input before the state x_2 starts significantly influencing the output as time progresses.

Appendix

Listing 1: Exercise 1.1

```
F1 = (3*s**2 + 5*s) / (s**3 + 6*s**2 + 11*s + 6)
f1 = inverse_laplace_transform(F1, s, t)

display(Math('F_1(s)␣=␣' + latex(F1)))
print()
display(Math('f_1(t)␣=␣' + latex(sp.simplify(f1))))
```

Listing 2: Exercise 1.2

```
F2 = (s**2 + 2*s + 1) / (s+2)**3
F2_partial = sp.apart(F2)
f2 = inverse_laplace_transform(F2_partial, s, t)

display(Math('F_2(s)␣=␣' + latex(F2)))
print()
display(Math('f_2(t)␣=␣' + latex(sp.simplify(f2))))
```

Listing 3: Exercise 1.3

```
F3 = (2*s + 3) / (s**3 + 6*s**2 + 21*s + 26)
f3 = inverse_laplace_transform(F3, s, t)

display(Math('F_3(s)␣=␣' + latex(F3)))
print()
display(Math('f_3(t)␣=␣' + latex(sp.simplify(f3))))
```

Listing 4: Exercise 1.4

```
F4 = (1 + 2*sp.exp(-s)) / (s**2 + 3*s + 2)
f4 = inverse_laplace_transform(F4, s, t)

display(Math('F_4(s)␣=␣' + latex(F4)))
print()
display(Math('f_4(t)␣=␣' + latex(sp.expand(f4))))
```

Listing 5: Exercise 2.1

```
num = np.array([1., 1.]) # Numerator coefficients
den = np.array([2., 3., 1.]) # Denominator coefficients
G = signal.TransferFunction(num, den)

# Generate impulse response
t_imp, imp_response = signal.impulse(G)

plt.figure(figsize=(10,5))
plt.plot(t_imp, imp_response)
plt.title('Impulse␣Response␣of␣G(s)')
plt.xlabel('Time')
plt.ylabel('Magnitude')
```

```python
plt.grid(True)

# Generate step respose
t_step, step_response = signal.step(G)

plt.figure(figsize=(10,5))
plt.plot(t_step, step_response)
plt.title('Step Response of G(s)')
plt.xlabel('Time')
plt.ylabel('Magnitude')
plt.grid(True)
```

Listing 6: Exercise 2.2

```python
plt.figure(figsize=(10,5))
alphas = [-1, 0, 5, 7]
for alpha in alphas:
  num = [alpha, 1]
  den = [2, 3, 1]
  G = signal.TransferFunction(num, den)
  # Computing the step response of the current G(s)
  t_step, step_response = signal.step(G)
  plt.plot(t_step, step_response, label=fr'$\alpha$ = {alpha}')

plt.title('Step Response of G(s)')
plt.xlabel('Time')
plt.ylabel('Magnitude')
plt.grid(True)
plt.legend()
plt.show()
```

Listing 7: Exercise 2.3

```python
zeros = [-1/alpha for alpha in alphas if alpha !=0]
plt.axvline(x=0, c='k') # y-axis
plt.axhline(y=0, c='k') # x-axis
plt.plot(zeros, [0]*len(zeros), 'o', label='zeros')
plt.plot([-1, -0.5], [0, 0], 'rx', label='poles')
plt.title('Pole-zero plot')
plt.xlabel('Real axis')
plt.ylabel('Imaginary axis')
plt.legend()
plt.grid()
plt.show()
```

Listing 8: Exercise 3.1

```python
time = np.linspace(0,15,100)
exp_1 = time*np.exp(-time)
exp_2 = np.exp(-time)

plt.figure(figsize=(10,5))
plt.plot(time, exp_1, label=r'$te^{-t}$')
plt.plot(time, exp_2, label=r'$e^{-t}$')
```

```
plt.title('System modes')
plt.xlabel('Time')
plt.ylabel('Magnitude')
plt.legend()
plt.grid()
plt.show()
```

Listing 9: Exercise 3.2

```
Y = sp.Function('Y')(s)
U = sp.Function('U')(s)

# Constants in the differential equation
a2, a1, a0, b2, b1, b0 = 1, 2, 1, 4, 15, 19

# Symbols for initial conditions
y0, dy0 = symbols('y_0 \dot{y}_0')

# Equation representing the differential equation in the Laplace domain
equation = sp.Eq(a2*(s**2*Y - s*y0 - dy0) + a1*(s*Y - y0) + a0*Y, U *(b2*
    s**2 + b1*s + b0))

# Solving the Laplace domain equation for Y(s)
Y_s = sp.solve(equation, Y)[0]

# Display Y(s) with the forced and free response
display(equation)
print()
display(Math('Y(s) = ' + latex(Y_s)))

real_heaviside = {Heaviside(t): Heaviside(t,1)}
# Substitute initial conditions and U = 0 for the free response
Y_s_free = Y_s.subs({y0: 1, dy0: 1, U: 0})
y_t_free = inverse_laplace_transform(Y_s_free, s, t)
display(Math('Y(s)_{free} = ' + latex(Y_s_free)))
print()
display(Math('y(t)_{free} = ' + latex(y_t_free)))

# Plot free response
time = np.linspace(0, 15, 100)
y_t_values = [y_t_free.subs({**real_heaviside, t: value}) for value in
    time]

plt.figure(figsize=(10,5))
plt.plot(time, y_t_values)
plt.title('Free Response')
plt.xlabel('Time')
plt.ylabel('Magnitude')
plt.grid(True)
plt.show()
```

Listing 10: Exercise 3.3

```
# Substitute U = step input and initial conditions to zero for the forced
    response
```

```
Y_s_forced = Y_s.subs({U: 1/s, y0: 0, dy0: 0})
y_t_forced = inverse_laplace_transform(Y_s_forced, s, t)
display(Math('Y(s)_{forced}␣=␣' + latex(Y_s_forced)))
print()
display(Math('y(t)_{forced}␣=␣' + latex(y_t_forced)))

# Plot forced response
time = np.linspace(0, 15, 100)
y_t_values = [y_t_forced.subs({**real_heaviside, t: value}) for value in
    time]

plt.figure(figsize=(10,5))
plt.plot(time, y_t_values)
plt.title('Forced␣Response')
plt.xlabel('Time')
plt.ylabel('Magnitude')
plt.grid(True)
plt.show()
```

Listing 11: Exercise 3.4

```
# Substitute the initial conditions and step input in Y(s)
Y_s_final = Y_s.subs({U: 1/s, y0: 1, dy0: 1})
y_t_final = inverse_laplace_transform(Y_s_final, s, t)
display(Math('Y(s)␣=␣' + latex(Y_s_final)))
print()
display(Math('y(t)␣=␣' + latex(y_t_final)))

# Plot response (free + forced)
time = np.linspace(0, 15, 100)
y_t_values = [y_t_final.subs({**real_heaviside, t: value}) for value in
    time]

plt.figure(figsize=(10,5))
plt.plot(time, y_t_values)
plt.title('Response␣y(t)␣to␣a␣Unit␣Step␣Input')
plt.xlabel('Time')
plt.ylabel('Magnitude')
plt.yticks([i for i in range(5,20,2)])
plt.grid(True)
plt.show()
```

Listing 12: Exercise 4.1

```
A = np.array([[0, 1], [-5, -2]])
B = np.array([[0], [2]])
C = np.array([[0, 1]])
D = np.array([[1]])

# Convert state-space representation to transfer function
num, den = signal.ss2tf(A, B, C, D)
G_s = signal.TransferFunction(num, den)
G_s_symbolic = (num[0,0]*s**2 + num[0,1]*s + num[0,2]) / (den[0]*s**2 +
    den[1]*s + den[2])
```

```
print(G_s)
print()
display(Math('G(s)␣=␣' + latex(G_s_symbolic)))
```

Listing 13: Exercise 4.2

```
u = sp.Function('u')(t)
y = sp.Function('y')(t)
diff_equation = sp.Eq(den[0]*sp.diff(y, t, t) + den[1]*sp.diff(y, t) +
    den[2]*y,
                num[0,0]*sp.diff(u,t,t) + num[0,1]*sp.diff(u,t) + num
                    [0,2]*u)
diff_equation
```

Listing 14: Exercise 4.3

```
import control

# finding the response (symbolic)
U_s = 1 / (s+3)
Y_s = G_s * U_s
y_t = inverse_laplace_transform(Y_s, s, t)
display(Math('Y(s)␣=␣' + latex(Y_s)))
print()
display(Math('y(t)␣=␣' + latex(sp.simplify(y_t))))

# Finding the states (symbolic)
A_s = sp.Matrix(A)
B_s = sp.Matrix(B)
I = sp.eye(A.shape[0]) # Identity matrix
alpha = (s * I - A_s).inv() * B
X_s_array = alpha * U_s

# Converting states variables from Laplace to time domain
X1_s = X_s_array[0]
X2_s = X_s_array[1]
x1_t = inverse_laplace_transform(X1_s, s, t)
x2_t = inverse_laplace_transform(X2_s, s, t)

display(Math('X(s)␣=␣' + latex(sp.simplify(X_s_array))))
print()
display(Math('X_1(s)␣=␣' + latex(sp.simplify(X1_s))))
print()
display(Math('X_2(s)␣=␣' + latex(sp.simplify(X2_s))))
print()
display(Math('x_1(t)␣=␣' + latex(sp.simplify(x1_t))))
print()
display(Math('x_2(t)␣=␣' + latex(sp.simplify(x2_t))))
print()

import control
# Create the state-space model
time = np.linspace(0, 10, 500)
```

```
sys = control.ss(A, B, C, D)
u = np.exp(-3 * time) * np.heaviside(time, 1)

# Simulate the response
_, yout, xout = control.forced_response(sys, T=time, U=u, return_x=True)

# Plotting
plt.figure(figsize=(14, 6))

# Plot the state evolution
plt.subplot(1, 2, 1)
plt.plot(time, xout.T)
plt.title('State Evolution')
plt.xlabel('Time (s)')
plt.ylabel('Magnitude')
plt.legend(['x1', 'x2'])
plt.grid()

# Plot the output evolution
plt.subplot(1, 2, 2)
plt.plot(time, yout)
plt.title('Output Evolution')
plt.xlabel('Time (s)')
plt.ylabel('Magnitude')
plt.grid()

plt.tight_layout()
plt.show()
```

References

[GRS01] B. Girod, R. Rabenstein, and A. Stenger. *Signals and Systems*. Wiley, 2001.

[Bro03] R. Bronson. *Schaum's Easy Outline of Differential Equations*. McGraw-Hill Education, 2003.

[LG05] B.P. Lathi and Roger A Green. *Linear systems and signals*. Vol. 2. Oxford University Press New York, 2005.

[Nis11] N.S. Nise. *Control Systems Engineering, Sixth*. John Wiley & Sons, Incorporated, 2011.

[Baz21] Dr. Trefor Bazett. *How to Solve Constant Coefficient Homogeneous Differential Equations*. 2021. URL: https://youtu.be/zn06v-8pvXo?si=jMbT8p4xKmdVtd0I.